Towards Continuous Subject Identification Using Wearable Devices and Deep CNNs

João Nadas*, Mona Jaber†, Sven van den Berghe‡, and Muhammad Imran*
*Communication, sensing and imaging (CSI) group, School of Engineering, University of Glasgow, U.K.
j.battitsella-nadas.1@research.gla.ac.uk and muhammad.imran@glasgow.ac.uk
†School of Electronic Engineering and Computer Science (EECS), Queen Mary University of London, U.K. m.jaber@qmul.ac.uk
‡Fujitsu Laboratories of Europe, Hayes Park, Middlesex, UK sven.vandenberghe@uk.fujitsu.com

***Abstract*—Subject identification has several applications. In transportation companies, knowing who is driving their vehicles might prevent theft or other ill-intended actions. On the other hand, privacy concerns, and the respective legislation, hinder the applicability of several traditional recognition techniques based on invasive technologies, such as video cameras. Moreover, in order to keep the driver's distractions to a minimum, this technologies must be non-disruptive, that is, they must be able to identify the subject seamlessly without them taking any action. In this context, we propose using deep learning applied to smart watch data for recognizing the person driving a vehicle based on a training set. Our proposal relies on the possibility of using transfer learning to avoid long training sessions for new drivers and to deliver a solution which can be deployed in practice. In this paper, we describe the convolutional neural network used in the solution and present results according to a real data-set collected by us, achieving accuracies ranging from 75 to 94%.**

I. Introduction

According to the most recent transport statistics issued by the Department for Transport in Great Britain, 78% of goods are moved by land transport covering 18.6 Billion kilometers on a domestic scale and which required 2.4 Million powered vehicles traveling to/from continental Europe in 2017[1]. It is imperative for the drivers' safety as well as for the insurance liability that the identity of the driver is continuously authenticated throughout the journey. The safety of the driver is highly critical during overnight long-haul journeys where there is a risk of hijackers or other malicious interventions. There were 2880 reported cargo thefts in Europe, Middle East, and Africa (EMEIA) countries in 2015 with a total of €54,986,504 combined loss; 2114 of these crimes involved thefts from vehicles. Moreover, the recent truck-driven attacks[2] in Tokyo (2019), London (2017), Stockholm (2017), Berlin (2016), and Nice (2016) have all been carried with stolen vehicles and have raised real concern of causing deliberate atrocities that could be prevented if truck drivers can be identified remotely. On the other hand, insurance policies advocate that the employee who is designated to drive a vehicle is indeed the driver during the entire journey and does not arrange for an unauthorized replacement. Moreover, transportation companies might use such a system to validate driving logs reported by drivers.

Continuous identification can be performed with high accuracy if a camera were placed in all such vehicles and the streaming video continuously analyzed [1]. However, such a system is intrusive and breaches policies related to privacy of employees and General Data Protection Regulations. To this end, there have been various efforts to remotely identify a driver using non-intrusive data collection such as Controller Area Network bus (CAN bus), wrist bands or similar.

For instance, authors in [2] achieve an accuracy of 73% in recognizing the identity of the driver by analyzing signals, such as the use of the accelerator pedal, brake pedal, vehicle velocity, and distance from the vehicle in front while drivers are asked to follow a car in a driving simulator. Authors in [3] and [4] model the pedal operation patterns of drivers using Gaussian mixture models and manage an accuracy of 76.8% of driver identification in a field test comprised of 276 drivers. A more recent work [5] couples the data from the CAN bus and the cell phone sensors to solve the driver recognition problem. The mobile phone sensor data is used as a "ground truth" for isolating events such as accelerating, turning, or breaking. However, the work was limited to two drivers and the accuracy reached only 60%. Authors in [6] attempt to identify the driver from data collected during a single turn from 12 sensors reporting every 0.1 second on the steering wheel parameters, velocity and acceleration, engine RPM, pedals data, torque, and throttle position with 50~79% accuracy. A different approach is used in [7] based on Extremely Randomized Tree or Extra-Tree that is used to analyze data from multiple sources (smart phone, vehicle, and virtual sensor) to identify a suspect replacing the legitimate driver with a promising 89% accuracy. Authors is [8] propose a solution to a different problem by collecting three-axes accelerometer and gyro-meter data from a remotely controlled car. Data from all six sensors is jointly analyzed using a Multi-View Convolutional Neural Network to accurately identify the car's manoeuvre such as acceleration, turning, and collision.

The varied efforts in this domain demonstrate further the need for inferring the driver's identity and/or behavior remotely. Moreover, the importance of jointly analyzing multiple sources in improving the accuracy of the classification is

[1]https://www.tapa-global.org/fileadmin/public/downloads/Membership/Tapa_Membership_Brochure_2018.pdf.

[2]https://www.express.co.uk/news/uk/794243/terror-attack-lorry-Westminster-Khalid-Masood-Stockholm-immobiliser-kill-switch.

apparent in most of the cited works. However, most of the experiments carried are mostly dominated by vehicle sensor data, thus, the classification may be over-fitted to the vehicle itself as opposed to the independent driver's signature. In our work, we extract the data features that capture the driver's identity in any vehicle and in any manoeuvre.

The goal of this work is to develop an affordable, reliable, non-intrusive, and seamless technology for continuous driver identification. In order to make it affordable, we rely on off-the-shelf embedded sensors, instead of trying to develop a solution using expensive—albeit precise—sensors such as Electrocardiogram (ECG) sensors. Furthermore, we need to have a transferable technology that can be used for any driver, or set of drivers. Moreover, we require a tested technology that produces high accuracy results in order to make it reliable while keeping false positives to a minimum. False positives would cause unnecessary disruptions to normal operation incurring inhibitive costs. More importantly, it would desensitize operators' reaction to raised alarms, whether true or false, which would compromise the safety and security measures. In addition, it must be non-intrusive to comply with regulations and make it acceptable to staff. Lastly, the method deployed needs to be seamless so drivers do not have to interrupt/compromise their driving to trigger the identification process and it should be continuous so any significant (i.e., not justified by communication delays) interruption is interpreted as an anomaly and would generate an alarm.

As demonstrated in the prior art towards subject identification, streaming cameras achieve excellent results; however, it is quite intrusive to permanently aim a camera at drivers as well as bandwidth expensive. The alternative of processing the video at the edge and only streaming the results of classification limits the usage of the camera and undermines the robustness of the solution. Other technologies are tightly coupled with the driving behaviour hence may be biased by the vehicle type and road conditions. Our proposed method avoids these shortfalls while producing reliable results. Firstly, the solution relies on detecting a person's signature from wearable device data, thus requires streaming real-time concise measurements over a secure wireless network (e.g., 3GPP technologies) to a secure server for continuous authentication. Wearable data qualifies as "personal data" under the EU Data Protection Directive 95/46/EC as it can be used to identify an individual). Moreover, if it can be used to monitor the physiological or mental health status of the subject, then it also qualifies as medical data according to European Data Protection Supervisor as well as Article 29 Working Party. As such, it is required to obtain an explicit and fully informed consent from the technology users. In our case, users are fleet drivers, often on long overnight hauls, and the technology proposed is the least intrusive that would guarantee their security throughout the journey. According to a recent survey conducted in the USA, more than 65% of workers have positive willingness to use wearable devices if used to improve their safety [9]. Although in this paper we only cover the solution for subject identification, wearable devices could also be used to monitor driver drowsiness and stress levels with minimal load on the wireless network.

To this end, we explore the usage of deep learning, a technology that has gained much traction in recent years due to its outstanding performance and which relies on automatic feature extraction, such that domain specific expert knowledge is not an impediment for developers. Furthermore, it enables the usage of transfer learning, which enables trained feature extractors to be reused. This is detailed in Section II. Moreover, this work looks at extracting the driver's signature from widely available sensors that collect heart rate (HR) information, as shown in Section III. The proposed method, presented in Section IV, is complemented with a post processing step to enhance the reliability of the identification results. Results presented in Section V are very promising and demonstrate the potential of deep learning to reveal the hidden signature of drivers from their biometric data.

II. Background

Machine learning (ML) is defined as designing computational systems to perform a task without explicitly programing all the decisions required for their completion [10]. There are several classes of ML algorithms, such as supervised learning, unsupervised learning, reinforcement learning, Markov models, heuristics, etcetera. In this paper, we are concerned with supervised learning, wherein the computer aims to learn from past experiences and then generalize what it has learn to make accurate predictions on unseen data [11]. In other words, our algorithm takes collected sensory information from test runs and learns how to classify the driver by training on this data.

There are several types of supervised learning techniques, such as Bayes' classifiers, neural networks, decision trees, K-nearest neighbors, and support vector machines, to name a few [10]. Supervised learning algorithms can be further categorized into shallow and deep learning classifiers [12]. Shallow classifiers rely on expert domain to design feature extractors from the available data. Feature extraction is a key research direction that requires a high level of expertise.

On the other hand, in deep learning, features are automatically extracted from inputs. One notable area where deep learning is often used in for image processing. Basically instead of trying to extract complex features from images, deep learning relies on feeding the entire image data to an algorithm and training it based on examples. Deep learning has gathered a lot of attention in the recent years due to its power and relative ease of implementation, which does not rely on expert domain knowledge. Recent publications show impressive performance when using deep learning for several challenging tasks, such as video recognition, image recognition, speech interpretation, and several others [12]. This massive spur in the recent years is tied to the fact that now researchers have access to much more powerful computers, which can learn several parameters in feasible time.

One of the most traditional examples of deep learning are deep convolutional neural networks (CNNs) for image

Fig. 1. The Ticwatch S from Mobvoi used to collect the data.

recognition. The idea in this case is to input the entire image—in as many channels are there are available, for instance an RGB image has 3 channels—and let the network figure out the features on its own. The first layers of the CNN is responsible for learning the features via convolutional kernels. That is, a filter is applied to the image considering only neighboring pixels and the output is fed into another convolutional layer to extract more coarse features from this filtered output. Using this succession of convolutional layers and training the network on labeled examples, the CNN learns the weights for the filter coefficients and automatically extract features from the image, without any image processing expertise.

Typically the output of the convolutional layers is then fed into a shallow classifier—some fully connected neural network layers in the case of CNNs—that, in turn, have the job of classifying the input based on the automatically extracted features. This is specially powerful as the weights previously learned in the training for the feature extraction part can be reused to classify entirely different types of data, thus potentially saving hundreds—if not thousands— of hours of computing time. AS such, only the shallow classifier needs to be trained for the new data. This is known as transfer learning, and it is an extremely powerful tool, as researchers from entirely different fields can use similar sets of weights to obtain good accuracy in classification, all relying on the fact that the feature extractor is good at recognizing common features in sets of inputs with the same dimensionality.

In our application, we are not trying to classify an image, but 1-D signals. Deep learning is still applicable, as the same idea still holds. That is, we want to input several time series, from different sensors, and gather features that relate the samples from each time series. In other words, we are capturing the relationship from neighboring samples in time and extracting, automatically, features from it, in order to classify the drivers.

III. Data Collection

Biometric and location data were collected from the drivers using wrist-worn smart watches. Fig. 1 shows the Ticwatch S from Mobvoi used in the data collection. Data were collected using a bespoke Android application installed onto the smart watches. The data were written to the smart watch internal memory as CSV files, stored on the smart watch and regularly downloaded to a PC using the Device File Explorer component of Android Studio. The bespoke application collected data from the linear acceleration, gyroscope, orientation, and heart-rate sensors. In addition, the global positioning system (GPS) unit was used to acquire location and speed of the watch.

TABLE I
Data Collection

Description	Driver I	Driver II
Number of trips	74	66
Total time driven (hours)	47.0	36.6
Average trip duration (s)	2289.56	1998.51
Total distance driven (km)	2228.68	1246.74
Average trip length (km)	30.11	18.89

The watches were generally operated in standalone mode, *i.e.* not connected to a phone. The GPS data were regularly reviewed to ensure that the position data maintained accuracy despite no connection to a phone to recalibrate the position and no evidence of location inaccuracy was seen.

Moreover, the drivers were requested to vary the wrist on which the smart watch was worn over the sampling drives. Changing wrists was not expected to affect the values collected for biometric data or vehicle location and speed, but, as the cars used were manual gear change, linear acceleration, orientation and gyroscope readings were expected to differ depending on which wrist was wearing the watch. The resulting dataset is separated into trips, such that each unique trip is labeled according to a trip identifier, the wrist where the watch was worn and the driver (anonymized) wearing the watch. Table I contains some summarized information about the data that has been collected for this experiment. Next we present a brief description of the output from each of the sensors and their respective sampling rates.

1) Linear Acceleration: A three dimensional vector indicating acceleration along each device axis, not including gravity. All values have units of $\mathrm{m/s^2}$. The average interval between reports over a typical journey was 28 milliseconds.

2) Gyroscope: All values are in $\mathrm{rad/s}$ and measure the rate of rotation around the device's local x, y and z axes. The average interval between reports over a typical journey was 28 milliseconds.

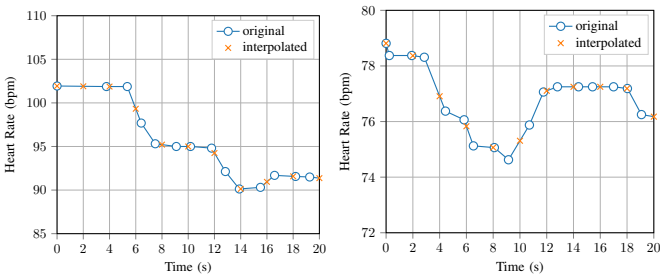
3) Orientation: The orientation sensors measures azimuth (angle between the magnetic north direction and the y-axis), pitch (rotation around x-axis), and roll (rotation around the y-axis) 3. The average interval between reports over a typical journey was 28 milliseconds.

4) Heart Rate: A photoplethysmogram (PPG) optical sensor, reporting HR in beats/min. The average interval between reports was 1275 milliseconds.

5) GPS: Location and speed were requested from the GPS unit once every second, however the actual average interval recorded for GPS data over was 1500 milliseconds.

IV. Implementation

As detailed in Section III, the available data captures the drivers' biometric information (i.e., the HR) and the behavioral information (Gyroscope and orientation that reflect the person's movement), in addition to the GPS. In this paper, we



(a) First example (b) Second example

Fig. 2. Examples of original data and interpolated output.

focus our work on the biometrics sensor data in order to extract the human-centric characteristics that identify the person under any condition or behavior. To this end, we propose to apply automatic feature extraction through deep learning and obtain the person's signature in three steps: Step one concerns the pre-processing of raw data to ready it for the deep learning phase; Step two consist of training and testing the CNN designed for this classification problem; Step three consists of post-processing the output of the CNN to improve its accuracy.

A. Pre-processing the data

After having selected which sensors to work with, we then proceeded with processing the sensory information such that it could be used in our deep learning model.

The first step is to align the starting and ending point of all the samples. This has to be done because when collecting the data, some sensors start and end at different time instants as the others. This is easily done by only taking data from where the sensors intersect.

Next, we match the sampling interval such that it is consistent across different measurements. This has to be done as the hardware does not keep a constant sampling rate and sometimes reports 2 samples one right after the other. Since we do not input time information into the CNN, if this is left untreated, the algorithm's performance would be hindered.

The strategy to match the sampling frequency is to set a predefined sampling rate and interpolate the data from the sensors such that all samples fall at the same time instants. In our case we chose 2 seconds, as this is a slightly lower frequency than the average for the HR sensor. Note that, for the CNN to make sense of the combined information from more than one sensor, all of them must be at the same sampling rate. We show on Fig. 2 an example of the HR data before and after the matching of the sampling rates. Note how if we took the original samples of Fig. 2a their value would align well with the interpolated data, however for Fig. 2b we can clearly see that everything would be displaced due to the 2 consecutive samples within the first 2 seconds.

The inputs for our CNN must have the same length and thus we must now split the data of each trip into segments. We must find a balance between the number of samples that our CNN has to work with in order to perform classification and the time it takes for us to identify who the driver is in a real-time deployment. Therefore, we chose to split the trips into 2 minute long segments, such that we can have a good response time and at the same time the CNN is fed with 60 samples for each sensor.

Moreover, because the CNN will be running on the real-time data constantly, we decided to split the trips in overlapping segments, such that we can run the classification every sampling interval (2 seconds). Note that, in the real-time situation, the sampling interval would also have to be respected, and if a sample is delayed (*i.e.* it is collected after the sampling instant has passed) classification would not be performed for that instant, while if a sample is collected to early (*i.e.* two consecutive samples are collected with almost no interval between them) it is discarded.

The next step is to split the segments into training and testing datasets. Since we are considering overlapping segments, the same trip should be entirely placed into either one of the datasets, to avoid misleading results. This happens because two consecutive segments share most of their samples (except for two) and therefore our classifier would have exceptional performance that cannot be replicated in a practical deployment. Thus, the trips are randomly assigned to either dataset with 85/15% ratio, respectively, for training and testing.

Once trips are properly split between datasets, the segments in each are scrambled and the data is normalized across the training set. Normalization is essential to avoid bias and also such that we can combine data from two different sensor sources with different scales. Normalization is performed according to

$$X_s = \frac{X_s - \mu_s}{\sigma_s}, \tag{1}$$

where the subscript s indicates the sensor type, X_s is the interpolated data, μ_s and σ_s indicate respectively the average and standard deviation for sensor s across the training set.

B. The Proposed CNN

The CNN we designed consists of three convolutional layers, followed by a pooling layer each in order to perform the feature extraction. The input layer has length 60 and is comprised of four channels, one for each axis of the accelerometer plus one for the HR data. After the last pooling layer, a flatten layer is added and then two hidden fully connected layer with 50 and 10 neurons, respectively. Finally the last layer is the output layer with one output per driver (two in our case).

Each convolutional layer has 60 filters with kernel size 8. In the first convolutional layer, this corresponds to capturing the relationship of HR and acceleration within 16 seconds. The activation function for the convolutional layers is the rectified linear unit (ReLU), as in [12]. The weights initializer for the kernel is a normal distribution with mean 0 and standard deviation [13]

$$\sigma_k = \frac{1}{\sqrt{\phi_i}}, \tag{2}$$

where ϕ_i is the number of inputs for that weight tensor.

The activation function for the fully connected hidden layer is hyperbolic tangent, to improve the convergence of stochastic gradient descent (SGD) algorithms [13]. To avoid over fitting of the dataset, we have introduced regularizes between each layer. The amount of regularization for each layer is 10^{-4}.

Even though we only have two drivers for this trial, in a practical deployment more drivers would be considered and therefore binary crossentropy is not adequate for the loss function. Therefore, on the output layer we have used a softmax activation function, and the categorical crossentropy loss function, accordingly. Lastly, the training method is the SGD, with learning rate 10^{-5} and momentum 0.9.

C. The post processing algorithm

In order to reduce the number of false-alarms in the real-time deployment we propose a post-processing algorithm. The idea is to apply a moving average filter on the measured data to eliminate false negatives which might appear surrounded by sequences of true positives. This way, higher frequencies are removed from the data, meaning that isolated negatives are removed from the result. After the filter is applied, a threshold is considered to decide if the entry is positive or negative.

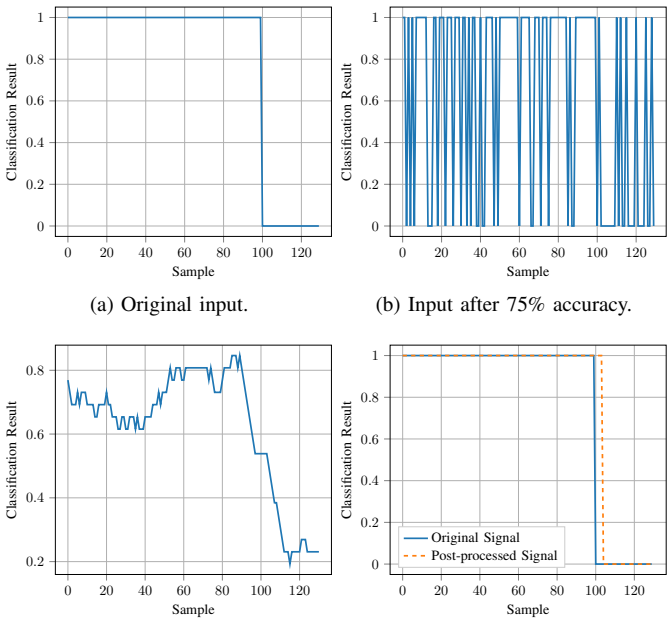
Basically we are sacrificing identification time in order to increase the reliability, such that when an alarm is fired it is most likely a true negative. This idea is illustrated in Fig. 3. In the illustration, the input represented in Fig. 3a, is a signal which represents positive or negative via zero and one. Fig. 3b shows the result of testing that signal with a 75% accurate classifier. After applying the moving average filter, the signal is much more closely related to the original input, as it can be observed in 3c. Finally, the result of applying the threshold is represented in 3d, alongside the original signal. As we can see, it was properly identified with some delay and high accuracy.

In a practical real-time deployment, wrongly identifying a driver might be costly due to the triggering of false-alarms which start safety procedures that might be detrimental to the transport company's operations. Since no system is perfect, we propose that, in a real-world deployment, when a driver is identified as not the one who is supposed to be driving, another mechanism could be used to prove the driver's identity, such as voice-recognition, since this does not require any additional hardware to be deployed with the driver and can provide a very strong level of confidence.

V. Results

In this Section we show the results of using the trained CNN on the test trips in order to determine between the two subjects. The parameters of the CNN and of the training are summarized in Table III. The aggregated results for the test trips are presented in the confusion matrix at Table II. As we can see there is a slight bias towards Driver I, and this will become apparent when we analyze the performance per trip. Despite this, the 94% and 75% accuracy results are promising.

The accuracy of predictions for each individual trips for both drivers is plotted in Fig. 4. The trips of Driver I are shown in Fig. 4a and the ones for Driver II are shown in Fig. 4b. As we



(a) Original input. (b) Input after 75% accuracy.

(c) Median filter applied. (d) Threshold applied.

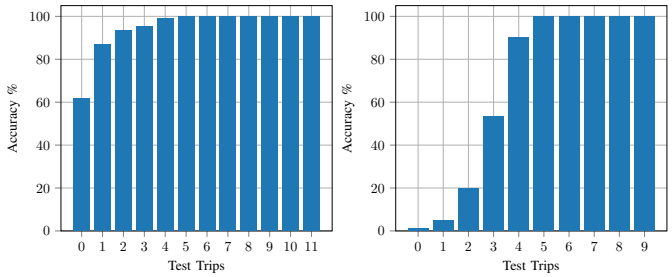
Fig. 3. The post processing algorithm. Note how after the low-pass filter and the decision threshold are applied the original signal can be recovered with a slight delay.

TABLE II
Confusion Matrix

		Predicted: Driver I	Predicted: Driver II	Accuracy
Actual	Driver I	**11457**	647	94.7 %
	Driver II	2885	**8809**	75.3 %

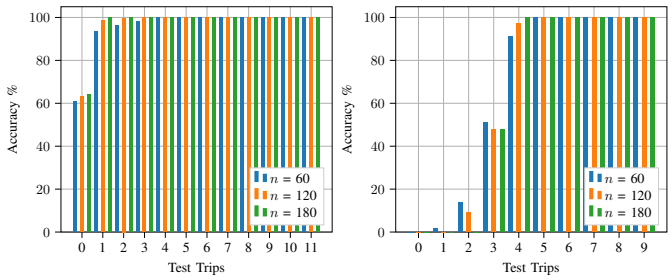
TABLE III
CNN Parameters

Parameter	Value
Layers	
L1 / L2 Regularizer	10^{-4}
Kernel Initializer	LeCun Normal
Bias Initializer	0
Convolutional	
Quantity	3
Filters per Layer	60
Kernel Size	8
Activation	ReLu
Fully Connected Hidden	
Quantity	2
Neurons per Layer	50 & 10
Activation	Hyperbolic Tangent
Output	
Neurons	2
Activation	Softmax
Training	
Batch Size	100
Number of Epochs	100
Training / Test ratio	85% / 15%
SGD Optimizer	
Learning Rate	10^{-5}
Momentum	0.9
Loss Function	Categorical Crossentropy



(a) Driver I (b) Driver II

Fig. 4. The accuracy of predictions for the test trips for both drivers.



(a) Driver I (b) Driver II

Fig. 5. The results of post-processing the data with a moving average filter, with length n, followed by a decision threshold.

can observe, the performance of the classifier is always above 80% for Driver I in all but 1 trip, which has 60% accuracy. Moreover, for Driver II the performance is above 80% for more than half the trips. For Driver II, however, we observe 4 trips with poor performance and in those cases an alarm would be triggered or a fail-safe mechanism would be required.

In order to improve our results, we applied the proposed post-processing algorithm for all the test trips. The results are shown in Fig. 5, with Fig 5a and Fig 5b plotting the results for Drivers I and II, respectively. Moreover, we tested the algorithm with 3 different lengths of moving average window (n). Note that the larger the value of n the longer it takes for the algorithm to compute the classification, and the more precise it is. This idea is well illustrated in Fig. 5a, where we are able to obtain 100% accuracy in all trips but one for $n \geq 120$ ($n = 120$ translates to 4 minutes of delay with a sampling rate of 2 seconds). However, this idea can only bring impressive improvements if the underlaying classification is already somewhat good. Note how Trip 0 for Driver I in Fig. 5a improved only slightly even for $n = 180$. The post-processing also brought improvements to Driver II, as it can be noted in Fig. 5b. However, because the underlaying classification was not as good, they only occur in one of the trips.

VI. Conclusion

The classification of drivers in a non-invasive manner is a challenging task and we have presented one alternative that works well in the dataset we have collected. More research is needed in order to provide more results with different datasets in order to consolidate the technology. However, we have shown in this paper that the approach is valid and with further improvements can be used in practical deployments for real-time classification of subjects. Moreover, we have shown how post-processing the data can improve the results, especially if the underlaying classification has a good performance (*i.e.* above 80% accuracy). One possible extensions of this work are: combining biometric and behavioral information with advanced deep learning techniques, such as imagification [14]. Furthermore, this work can also be extended by studying the transfer learning capabilities of the proposed method in order to use the obtained features with a different dataset.

Acknowledgment

This work was funded by Fujitsu Laboratories of Europe.

References

[1] J. Zeng, Y. Sun, and L. Jiang, "Driver distraction detection and identity recognition in real-time," in *Proc. Second WRI Global Congress Intelligent Systems*, vol. 3, Dec. 2010, pp. 43–46.

[2] T. Wakita, K. Ozawa, C. Miyajima, and K. Takeda, "Parametric versus non-parametric models of driving behavior signals for driver identification," in *Proc. International Conference on Audio-and Video-Based Biometric Person Authentication*. Springer, Jul. 2005, pp. 739–747.

[3] C. Miyajima, Y. Nishiwaki, K. Ozawa, T. Wakita, K. Itou, K. Takeda, and F. Itakura, "Driver modeling based on driving behavior and its evaluation in driver identification," *Proceedings of the IEEE*, vol. 95, no. 2, pp. 427–437, Feb. 2007.

[4] C. Miyajima, Y. Nishiwaki, K. Ozawa, T. Wakita, K. Itou, and K. Takeda, "Cepstral analysis of driving behavioral signals for driver identification," in *Proc. IEEE Int Conf. Acoustics Speech and Signal Processing*, vol. 5, May 2006, p. V.

[5] M. Van Ly, S. Martin, and M. M. Trivedi, "Driver classification and driving style recognition using inertial sensors," in *Proc. IEEE Intelligent Vehicles Symp. (IV)*, Jun. 2013, pp. 1040–1045.

[6] D. Hallac, A. Sharang, R. Stahlmann, A. Lamprecht, M. Huber, M. Roehder, R. Sosič, and J. Leskovec, "Driver identification using automobile sensor data from a single turn," in *Proc. IEEE 19th Int. Conf. Intelligent Transportation Systems (ITSC)*, Nov. 2016, pp. 953–958.

[7] P. H. L. Rettore, A. B. Campolina, A. Souza, G. Maia, L. A. Villas, and A. A. F. Loureiro, "Driver authentication in vanets based on intra-vehicular sensor data," in *Proc. IEEE Symp. Computers and Communications (ISCC)*, Jun. 2018, pp. 00 078–00 083.

[8] Y. Zhang, J. Li, Y. Guo, C. Xu, J. Bao, and Y. Song, "Vehicle driving behavior recognition based on multi-view convolutional neural network (MV-CNN) with joint data augmentation," *IEEE Transactions on Vehicular Technology*, p. 1, 2019.

[9] J. V. Jacobs, L. J. Hettinger, Y.-H. Huang, S. Jeffries, M. F. Lesch, L. A. Simmons, S. K. Verma, and J. L. Willetts, "Employee acceptance of wearable technology in the workplace," *Applied Ergonomics*, vol. 78, pp. 148 – 156, 2019. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0003687018306094

[10] P. V. Klaine, M. A. Imran, O. Onireti, and R. D. Souza, "A survey of machine learning techniques applied to self-organizing cellular networks," *IEEE Communications Surveys Tutorials*, vol. 19, no. 4, pp. 2392–2431, 2017.

[11] S. B. Kotsiantis, "Supervised machine learning: a review of classification techniques," *Informatica*, vol. 31, no. 3, p. 249, 2007.

[12] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[13] Y. A. LeCun, L. Bottou, G. B. Orr, and K.-R. Müller, *Efficient BackProp*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 9–48.

[14] S. Furuya, A. Sanaee, S. Georgescu, J. Townsend, B. Rasmussen, P. Chow, D. Snelling, and M. Goto, "Imagification technology and deep learning accelerating defect detection in non-destructive testing for wind turbine blades," *FUJITSU SCIENTIFIC & TECHNICAL JOURNAL*, vol. 55, no. 2, pp. 23–29, 2019.